ANL-7440 ANL-7440

Argonne National Laboratory

NUMERICAL INVERSION OF FINITE TOEPLITZ MATRICES AND VECTOR TOEPLITZ MATRICES

by

Erwin H. Bareiss

The facilities of Argonne National Laboratory are owned by the United States Government. Under the terms of a contract (W-31-109-Eng-38) between the U. S. Atomic Energy Commission, Argonne Universities Association and The University of Chicago, the University employs the staff and operates the Laboratory in accordance with policies and programs formulated, approved and reviewed by the Association.

Printed in the United States of America
Available from
Clearinghouse for Federal Scientific and Technical Information
National Bureau of Standards, U. S. Department of Commerce
Springfield, Virginia 22151
Price: Printed Copy $3.00; Microfiche $0.65

ANL-7440
Mathematics and
Computers (TID-4500)
AEC Research and
Development Report

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

NUMERICAL INVERSION OF FINITE TOEPLITZ MATRICES AND VECTOR TOEPLITZ MATRICES

by

Erwin H. Bareiss

Applied Mathematics Division

June 1968

TABLE OF CONTENTS

NUMERICAL INVERSION OF FINITE TOEPLITZ MATRICES AND VECTOR TOEPLITZ MATRICES

by

Erwin H. Bareiss

I. INTRODUCTION

Many problems of mathematical physics, statistics, and algebra lead to the problem of finding the inverse of finite *Toeplitz* or *Hankel* matrices. Well known are problems involving convolutions, integral equations with difference kernels, and least-square approximations by polynomials. Although there exists an abundant literature on the mathematical properties of Toeplitz matrices, there seem to be only a few references to the problem of numerical inversion.[1-4] The efficiencies of numerical methods involving Toeplitz or Hankel matrices are often judged under the assumption that the inversion of a Toeplitz matrix of order n requires of the order of n^3 multiplications. The purpose of this paper is to introduce a new method by which the exact inversion can be accomplished simply, using in the order of n^2 multiplications. Some efficient algorithms are given. Extension is made to vector Toeplitz matrices which occurred in the author's work.

II. INVERSION OF FINITE TOEPLITZ MATRICES

We present an algorithm to solve

$$Ax = c, \tag{2.1}$$

where A is a Toeplitz matrix and c a column vector denoted by

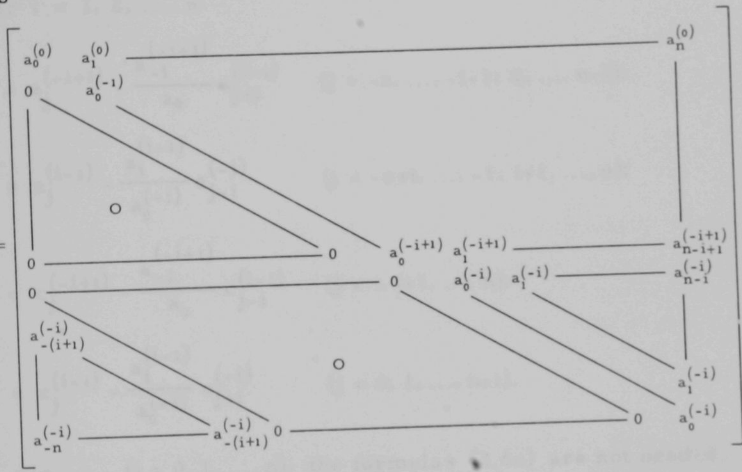
$$A = \begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_n \\ a_{-1} & a_0 & a_1 & \cdots & a_{n-1} \\ a_{-2} & a_{-1} & a_0 & \cdots & a_{n-2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{-n} & a_{-(n-1)} & a_{-(n-2)} & \cdots & a_0 \end{bmatrix}; \quad c = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}. \tag{2.2}$$

The basic idea is to transform (2.1) successively into

$$A^{(-1)}x = c^{(-1)};\ A^{(1)}x = c^{(1)};\ A^{(-2)}x = c^{(-2)};\ A^{(2)}x = c^{(2)};$$
$$\ldots;\ A^{(-n)}x = c^{(-n)};\ A^{(n)}x = c^{(n)}. \tag{2.3}$$

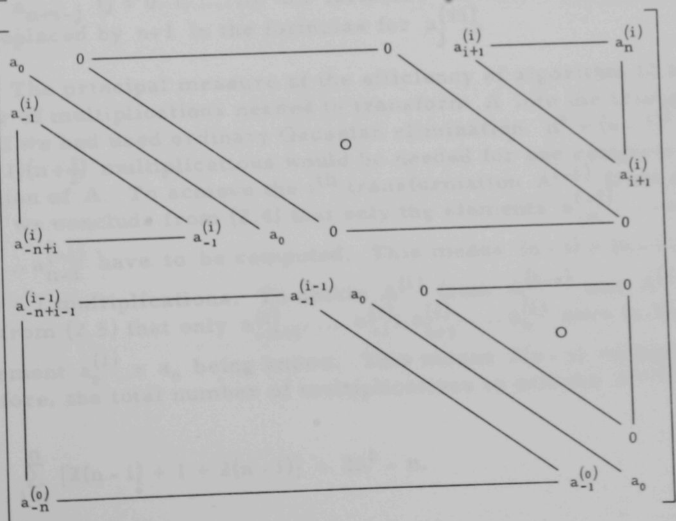
The matrices $A^{(-i)}$ have zero elements along the i subdiagonals below the main diagonal; the matrices $A^{(i)}$ have zero elements along the i superdiagonals above the main diagonal. Thus, $A^{(-n)}$ is an upper triangular matrix and $A^{(n)}$ a lower triangular matrix. The transformation $A^{(-i+1)} \Rightarrow A^{(-i)}$ affects only the rows i, i+1, ..., n; the transformation $A^{(i-1)} \Rightarrow A^{(i)}$ affects only the rows 0, 1, ..., n-i. Explicitly, $A^{(-i)}$ and $A^{(i)}$ assume the forms

$A^{(-i)} =$



(2.4)

$A^{(i)} =$



(2.5)

$$A^{(-1)}x = c^{(-1)};\ A^{(1)}x = c^{(1)};\ A^{(-2)}x = c^{(-2)};\ A^{(2)}x = c^{(2)};$$

$$\ldots;\ A^{(-n)}x = c^{(-n)};\ A^{(n)}x = c^{(n)} \qquad (2.3)$$

The matrices $A^{(-i)}$ have zero elements along the i subdiagonals below the main diagonal; the matrices $A^{(i)}$ have zero elements along the i superdiagonals above the main diagonal. Thus, $A^{(-n)}$ is an upper triangular matrix and $A^{(n)}$ a lower triangular matrix. The transformation $A^{(-i+1)} \Rightarrow A^{(-i)}$ affects only the rows i, i+1, ..., n; the transformation $A^{(i-1)} \Rightarrow A^{(i)}$ affects only the rows 0, 1, ..., n-1. Explicitly, $A^{(-i)}$ and $A^{(i)}$ assume the forms

$A^{(-i)} =$ (2.4)

$A^{(i)} =$  (2.5)

We note that the lower n-i+1 rows of $A^{(-i)}$ and the upper n-i+1 rows of $A^{(i)}$ again form (rectangular) Toeplitz matrices. Furthermore, $a_0 = a_0^{(1)} = \ldots = a_0^{(i)} = \ldots = a_0^{(n)}$. The basic algorithm for the transformations (2.3) is given by

$$a_j^{(0)} = a_j \qquad (j = -n, -n+1, \ldots, 0, \ldots, n);$$

$$c_j^{(0)} = c_j \qquad (j = 0, \ldots, n), \tag{2.6a}$$

and for $i = 1, 2, \ldots, n$:

$$a_j^{(-i)} = a_j^{(-i+1)} - \frac{a_{-i}^{(-i+1)}}{a_0} a_{j+i}^{(i-1)} \qquad (j = -n, \ldots, -i-1;\ 0, \ldots, n-i);$$

$$a_j^{(i)} = a_j^{(i-1)} - \frac{a_i^{(i-1)}}{a_0^{(-i)}} a_{j-i}^{(-i)} \qquad (j = -n+i, \ldots, -1;\ i+1, \ldots, n); \tag{2.6b}$$

$$c_j^{(-i)} = c_j^{(-i+1)} - \frac{a_{-i}^{(-i+1)}}{a_0} c_{j-i}^{(i-1)} \qquad (j = i, i+1, \ldots, n);$$

$$c_j^{(i)} = c_j^{(i-1)} - \frac{a_i^{(i-1)}}{a_0^{(-i)}} c_{j+i}^{(-i)} \qquad (j = 0, 1, \ldots, n-i). \tag{2.6c}$$

If $c_j = a_{n+1-j}$ $(j = 0, 1, \ldots, n)$, the formulas (2.6c) are not needed. Instead, +n is replaced by n+1 in the formulas for $a_j^{(\pm i)}$.

The principal measure of the efficiency of algorithm (2.6) is the number of multiplications needed to transform A into the triangular forms $A^{\pm n}$. If we had used ordinary Gaussian elimination, $n^2 + (n-1)^2 + \ldots + 1 = \frac{1}{3}n(n+1)(n+\frac{1}{2})$ multiplications would be needed for one complete triangularization of A. To achieve the i^{th} transformation $A^{(-i)}$ from $A^{(-i+1)}$ and $A^{(i-1)}$, we conclude from (2.4) that only the elements $a_{-n}^{(-i)}, \ldots, a_{-(i+1)}^{(-i)}$; $a_0^{(-i)}, \ldots, a_{n-i}^{(-i)}$ have to be computed. This means $(n-i) + (n-i+1) = 2(n-i) + 1$ multiplications. To obtain $A^{(i)}$ from $A^{(i-1)}$ and $A^{(-i)}$ we conclude from (2.5) that only $a_{-n+i}^{(i)}, \ldots, a_{-1}^{(i)}$; $a_{i+1}^{(i)}, \ldots, a_n^{(i)}$ have to be computed, the element $a_0^{(i)} = a_0$ being known. This means $2(n-i)$ multiplications. Therefore, the total number of multiplications to achieve $A^{(-n)}$ and $A^{(n)}$ is

$$\sum_{i=1}^{n} [2(n-i) + 1 + 2(n-i)] = 2n^2 - n.$$

We note that the lower $s-i+1$ rows of $A^{(-i)}$ and the upper $n-i$ rows of $A^{(i)}$ again form (rectangular) Toeplitz matrices. Furthermore, $a_0 = a_0^{(0)} = \ldots = a_0^{(i)} = \ldots = a_0^{(n)}$. The basic algorithm for the transformations (2.3) is given by

$$a_j^{(0)} = a_j \qquad (j = -s, -n+1, \ldots, 0, \ldots, n),$$

$$c_j^{(0)} = \quad (j = 0, \ldots, n), \qquad (2.6a)$$

and for $i = 1, 2, \ldots, n$:

$$a_j^{(-i)} = a_j^{(-i+1)} - \frac{a_{-i}^{(-i+1)}}{a_0} a_{i+j}^{(i-1)} \qquad (j = -n, \ldots, -i-1;\ 0, \ldots, n-i),$$

$$a_j^{(i)} = a_j^{(i-1)} - \frac{a_i^{(i-1)}}{a_0} a_{j-i}^{(-i+1)} \qquad (j = -n+1, \ldots, -1;\ i+1, \ldots, n), \qquad (2.6b)$$

$$c_j^{(-i)} = c_j^{(-i+1)} - \frac{a_{-i}^{(-i+1)}}{a_0} c_{i+j}^{(i-1)} \qquad (j = i, i+1, \ldots, n),$$

$$c_j^{(i)} = c_j^{(i-1)} - \frac{a_i^{(i-1)}}{a_0} c_{j-i}^{(-i+1)} \qquad (j = 0, 1, \ldots, n-i). \qquad (2.6c)$$

If $c_j = a_{n+1-j}$ $(j = 0, 1, \ldots, n)$, the formulas (2.6c) are not needed. Instead $+n$ is replaced by $n+1$ in the formulas for $a_j^{(\pm i)}$.

The principal measure of the efficiency of algorithm 1 is the number of multiplications needed to transform A into the triangular forms $A^{(n)}$. If we had used ordinary Gaussian elimination, $n^3 + \frac{1}{2}n(n+1)(n+\frac{1}{2})$ multiplications would be needed for the complete triangularization of A. To achieve the i^{th} transformation $A^{(-i)}$ from $A^{(-i+1)}$ and $A^{(i-1)}$, we conclude from (2.3) that only the elements $a_{-n}^{(-i)}, \ldots, a_{-i-1}^{(-i)}$, $a_0^{(-i)}, \ldots, a_{n-i}^{(-i)}$ have to be computed. This means $(n-i) + (n-i+1) = 2(n-i) + 1$ multiplications. To obtain $A^{(i)}$ from $A^{(i-1)}$ and $A^{(-i+1)}$ we conclude from (2.3) that only $a_{-n+1}^{(i)}, \ldots, a_{-1}^{(i)}, a_{i+1}^{(i)}, \ldots, a_n^{(i)}$ have to be computed, the element $a_0^{(i)} = a_0$ being known. This means $2(n-i)$ multiplications. Therefore, the total number of multiplications to achieve $A^{(-n)}$ and $A^{(n)}$ is

$$\sum_{i=1}^{n} [2(n-i) + 1 + 2(n-i)] = 2n^2 - n.$$

To compute $c_1^{(\pm 1)}, \ldots, c_1^{(\pm n)}$, we need an additional $n^2 + n$ multiplications in the general case, $2n$ multiplications in the special case $c_j = a_{n+1-j}$. With Gaussian elimination $\frac{1}{2}(n^2+n)$ multiplications would be needed.

The solution of

$$A^{(-n)}x = c^{(-n)} \quad \text{or} \quad A^{(n)}x = c^{(n)}$$

requires, as in Gaussian elimination, $\frac{1}{2}(n^2+n)$ multiplications. However, if we take rows $0, 1, \ldots, \left[\frac{n-1}{2}\right]$ of $A^{(n)}$ and rows $\left[\frac{n+1}{2}\right], \ldots, n$ of $A^{(n-1)}$ to solve for x, we need only

$$1 + 2 + \ldots + \left[\frac{n-1}{2}\right] + 1 + 2 + \ldots + \left[\frac{n}{2}\right] = \begin{cases} \frac{1}{4}(n^2 - 1) & \text{(n odd)} \\ \frac{1}{4}n^2 & \text{(n even)} \end{cases}$$

multiplications, a saving of $\left[\frac{n+1}{2}\right]\left[\frac{n+2}{2}\right]$ multiplications, i.e., more than one-half. The number of quotients to be computed in (2.6) is only of order n. [Note that $(1/a_0)\, a_i^{(-i+1)}$ $(i=1, \ldots, n)$, can be obtained with one division and n multiplications.] Thus, x in (2.1) can be obtained with no more than

$$(2n^2 - n) + (n^2 + n) + \frac{1}{4}n^2 = 3\frac{1}{4}n^2 \tag{2.7}$$

multiplications, using Eqs. (2.6). Gaussian elimination would require $\frac{1}{6}[n(n+1)(2n+7)] = \frac{1}{3}n^3 + \frac{3}{2}n^2 + \frac{7}{6}n$ multiplications. Algorithm (2.6) is therefore always recommended when $n > 4$.

In the basic algorithm (2.6) all pivotal elements a_0 and $a_0^{(-i)}$ are implicitly assumed to be different from zero. We assume now that of the $2n + 1$ elements in (2.2),

$$a_{-\mu+1} = \ldots = a_{\nu-1} = 0 \qquad (\mu, \nu > 0;\ \mu + \nu \leq n). \tag{2.8}$$

If $\mu + \nu = n + 1$, the matrix A is trivially reduced to a direct sum of two triangular matrices and needs no further transformation.

One method to triangularize (2.2) if (2.8) holds is to let a_ν and a_ν^{-1} take the roles of the pivotal elements. The formulas (2.6b) are then replaced by

$$a_j^{(-i)} = \begin{cases} a_j^{(0)} & \text{for } i = 1, \ldots, \mu+\nu-1; \\[2ex] a_j^{(-i+1)} - \dfrac{a_{\nu-i}^{(-i+1)}}{a_\nu}\, a_{j+i}^{(i-1)} & \text{for } i = \mu+\nu, \ldots, n; \end{cases}$$

and

$$j = \begin{cases} -n, \ldots, \nu-i-1;\ \nu, \ldots, n-i & \text{if } i \le n-\nu; \\ -n, \ldots, \nu-i-1 & \text{if } i > n-\nu; \end{cases} \tag{2.9}$$

$$a_j^{(i)} = \begin{cases} a_j^{(i-1)} - \dfrac{a_{\nu+i}^{(i-1)}}{a_\nu^{(-i)}}\, a_{j-i}^{(-i)} & \text{for } i = 1, \ldots, n-\nu; \\[2ex] a_j^{(n-\nu)} & \text{for } i = n-\nu+1, \ldots, n; \end{cases}$$

$$(j = -n+i, \ldots, \min(i-\mu, \nu-1);\ \nu+i+1, \ldots, n, \text{ but } j \le n).$$

For $\mu = 1$ and $\nu = 0$, these formulas become (2.6b). After n iterations, the matrix (2.2), under the conditions (2.8), takes the forms

$$A^{(-n)} = \left[\begin{array}{ccc|ccc|ccc}
0 & \cdots & 0 & & & & a_\nu^0 & \cdots\cdots & a_{2\nu+\mu-1}^{(0)} \;\cdots\; a_n^{(0)} \\
\vdots & & \vdots & 0 & & & & \ddots & \vdots \\
0 & \ddots & & \vdots & \ddots & & & & \vdots \\
a_{-\mu}^{(0)} & \ddots & 0 & & & & & & \vdots \\
a_{-\mu-\nu+1}^{(0)} & \cdots & a_{-\mu}^{(0)} & 0 & \cdots & 0 & a_\nu^{(0)} & \cdots\cdots & a_{n-\mu-\nu+1}^{(0)} \\
a_{-\mu-\nu}^{(-\mu-\nu)} & \cdots & a_{-\mu-1}^{(-n-\nu)} & 0 & \cdots & 0 & a_\nu^{(-\mu-\nu)} & \cdots & a_{n-\mu-\nu}^{(-\mu-\nu)} \\
\vdots & & \vdots & \vdots & \ddots & & & \ddots & \vdots \\
a_{\nu-n}^{(\nu-n)} & \cdots & a_{2\nu-n+1}^{(\nu-n)} & 0 & \cdots & 0 & & & a_\nu^{(\nu-n)} \\
a_{\nu-n+1}^{(\nu-n-1)} & \cdots & a_{2\nu-n}^{(\nu-n-1)} & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\
\vdots & & \vdots & \vdots & & & & & \vdots \\
a_{-n}^{(-n)} & \cdots & a_{\nu-n+1}^{(-n)} & 0 & \cdots & \cdots & \cdots & \cdots & 0
\end{array}\right] \tag{2.10}$$

$$A^{(n)} = \left[\begin{array}{ccccccccc}
a_0^{(n-\nu)} & \cdots & a_{\nu-1}^{(n-\nu)} & a_\nu & & & 0 & \cdots & 0 \\
\vdots & & \vdots & \vdots & \ddots & & & \ddots & \vdots \\
a_{-\nu+1}^{(n-\nu)} & \cdots & a_0^{(n-\nu)} & a_1^{(n-\nu)} & \cdots & a_\nu & 0 & \cdots & 0 \\
a_{-\nu}^{(n-\nu)} & \cdots & a_{-1}^{(n-\nu)} & a_0^{(n-\nu)} & \cdots & a_{\nu-1}^{(n-\nu)} & a_\nu & \cdots & 0 \\
\vdots & & \vdots & \vdots & & \vdots & & \ddots & \vdots \\
a_{\nu+\mu-n-1}^{(\nu+\mu-1)} & \cdots & a_{2\nu+\mu-n-2}^{(\nu+\mu-1)} & a_{2\nu+\mu-n-1}^{(\nu+\mu-1)} & \cdots & a_{\nu-1}^{(\nu+\mu-1)} & a_\nu & \ddots & 0 \\
\vdots & & \vdots & \vdots & & \vdots & 0 & \ddots & a_\nu \\
\vdots & & \vdots & \vdots & & \vdots & \vdots & \ddots & 0 \\
a_{-n}^{(0)} & \cdots & a_{\nu-n+1}^{(0)} & a_{\nu-n}^{(0)} & \cdots & a_{-\mu}^{(0)} & 0 & \cdots & 0
\end{array}\right] \tag{2.11}$$

where in $A^{(n)}$ we have $a_\nu^{(0)} = \ldots = a_\nu^{(n-\nu)} = a_\nu$. To complete the triangularization we have to triangularize the square matrix of order ν formed by the last ν rows and first ν columns of $A^{(-n)}$.

III. INVERSION OF SYMMETRIC TOEPLITZ MATRICES

Let the algorithm (2.6) be replaced by

$$a_j^{(0)} = a_j, \ (-n \le j \le n); \ c_j^{(0)} = c_j, \ (0 \le j \le n), \tag{3.1a}$$

and for i = 1, 2, ..., n:

$$a_j^{(-i)} = a_j^{(-i+1)} - \frac{a_{-i}^{(-i+1)}}{a_0^{(i-1)}} a_{j+i}^{(i-1)} \qquad (j = -n, \ldots, -i-1; \ 0, \ldots, n-i); \tag{3.1b}$$

$$a_j^{(i)} = a_j^{(i-1)} - \frac{a_i^{(i-1)}}{a_0^{(-i+1)}} a_{j-i}^{(-i+1)} \qquad (j = -n+i, \ldots, 0; \ i+1, \ldots, n);$$

$$c_j^{(-i)} = c_j^{(-i+1)} - \frac{a_{-i}^{(-i+1)}}{a_0^{(i-1)}} c_{j-i}^{(i-1)} \qquad (j = i, i+1, \ldots, n); \tag{3.1c}$$

$$c_j^{(i)} = c_j^{(i-1)} - \frac{a_i^{(i-1)}}{a_0^{(-i+1)}} c_{j+i}^{(-i+1)} \quad (j = 0, 1, \ldots, n-i).$$

This algorithm is symmetric, and $a_j^{(i)}$ does not depend on $a_j^{(-i)}$. It is less efficient than (2.6) since $a_0^{(i)}$ $(i = 1, \ldots, n)$ must be computed. However, if A in (2.2) is symmetric, i.e.,

$$a_j = a_{-j} \quad (j = 1, 2, \ldots, n), \tag{3.2}$$

it follows by induction that

$$a_j^{(i)} = a_{-j}^{(-i)}. \tag{3.3}$$

Assume $a_j^{(i-1)} = a_{-j}^{(-i+1)}$ to be true; then by (3.1b)

$$a_{-j}^{(-i)} = a_{-j}^{(-i+1)} - \frac{a_{-i}^{(-i+1)}}{a_0^{(i-1)}} a_{-j+i}^{(i-1)} = a_j^{(i-1)} - \frac{a_i^{(i-1)}}{a_0^{(-i+1)}} a_{j-i}^{(-i+1)} = a_j^{(i)}.$$

Since by (3.2) the assumption is true for $i = 0$, (3.3) is proved. The effect is that the element in the j^{th} row and k^{th} column of $A^{(i)}$ in (2.5) is equal to the element of the $(n+1-j)^{th}$ row and $(n+1-h)^{th}$ column of $A^{(-i)}$ in (2.4). The algorithm (3.1) reduces therefore for *symmetric Toeplitz matrices* to

$$a_j^{(0)} = a_{|j|} (-n \leq j \leq n); \; c_j^{(0)} = c_j, \; (0 \leq j \leq n); \tag{3.4a}$$

$$a_j^{(i)} = a_j^{(i-1)} - \frac{a_i^{(i-1)}}{a_0^{(i-1)}} a_{i-j}^{(i-1)}; \quad (j = -n+i, \ldots, 0; i+1, \ldots, n) \tag{3.4b}$$

$$a_j^{(-i)} = a_{-j}^{(i)};$$

$$c_j^{(i)} = c_j^{(i-1)} - \frac{a_i^{(i-1)}}{a_0^{(i-1)}} c_{j+i}^{(-i+1)} \quad (j = 0, 1, \ldots, n-i);$$

$$c_j^{(-i)} = c_j^{(-i+1)} - \frac{a_i^{(i-1)}}{a_0^{(i-1)}} c_{j-i}^{(i-1)} \quad (j = i, i+1, \ldots, n). \tag{3.4c}$$

This algorithm takes $n(n-1)$ less multiplications than (2.6), and n^2 less multiplications than (3.1), for the computation of the $a_j^{(\pm i)}$ $(i = 1, \ldots, n)$.

Since symmetric Toeplitz matrices are centrosymmetric matrices, the computational work can also be reduced as follows.

Let $a_j = a_{-j}$ in (2.2), subtract row n-j from row $j\left(j = 0, 1, \ldots, \left[\frac{n}{2}\right]\right)$ in (2.1), and simplify to obtain for the relation

$$\begin{bmatrix} a_0 - a_n & a_1 - a_{n-1} & a_2 - a_{n-2} & \cdots & a_{\left[\frac{n}{2}\right]} - a_{n-\left[\frac{n}{2}\right]} \\ a_1 - a_{n-1} & a_0 - a_{n-2} & a_1 - a_{n-3} & \cdots & \cdots \\ a_2 - a_{n-2} & a_1 - a_{n-3} & a_0 - a_{n-4} & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{\left[\frac{n}{2}\right]} - a_{n-\left[\frac{n}{2}\right]} & \cdots & \cdots & \cdots & a_0 - a_{n-2\left[\frac{n}{2}\right]} \end{bmatrix} \begin{bmatrix} x_0 - x_n \\ x_1 - x_{n-1} \\ x_2 - x_{n-2} \\ \cdots \\ x_{\left[\frac{n}{2}\right]} - x_{n-2\left[\frac{n}{2}\right]} \end{bmatrix} = \begin{bmatrix} c_0 - c_n \\ c_1 - c_{n-1} \\ c_2 - c_{n-2} \\ \cdots \\ c_{\left[\frac{n}{2}\right]} - c_{n-2\left[\frac{n}{2}\right]} \end{bmatrix}. \tag{3.5a}$$

Similarly, add row n-j to row $j\left(j = 0, 1, \ldots, \left[\frac{n}{2}\right]\right)$ to obtain the relation

$$\begin{bmatrix} a_0 + a_n & a_1 + a_{n-1} & a_2 + a_{n-2} & \cdots \\ a_1 + a_{n-1} & a_0 + a_{n-2} & a_1 + a_{n-3} & \cdots \\ a_2 + a_{n-2} & a_1 + a_{n-3} & a_0 + a_{n-4} & \cdots \\ \cdots & \cdots & \cdots & \cdots \end{bmatrix} \begin{bmatrix} x_0 + x_n \\ x_1 + x_{n-1} \\ x_2 + x_{n-2} \\ \cdots \end{bmatrix} = \begin{bmatrix} c_0 + c_n \\ c_1 + c_{n-1} \\ c_2 + c_{n-2} \\ \cdots \end{bmatrix}. \tag{3.5b}$$

Therefore, the symmetric problem (2.1) has been reduced to two symmetric problems of order $(n+1)/2$ if n is odd, and of orders $n/2$ and $(n/2) + 1$ if n is even.

If n is odd, the solution of both (3.5a) and (3.5b) for $x_0, \ldots, x_n$ requires

$$\frac{(n^2 - 1)(n + 15)}{24}$$

multiplications plus $(n+1)$ multiplications by 2 (shift operations!), needed to obtain $x_0, \ldots, x_n$ from $x_0 \pm x_n, \ldots, x_{\frac{n-1}{2}} \pm x_{\frac{n+1}{2}}$. This compares with

$$\frac{n(n + 1)(n + 8)}{6}$$

multiplications using "symmetric" Gaussian elimination on the original matrix, and to

$$\frac{9n^2 - 1}{4} + n$$

multiplications to solve the problem by (3.4). It follows that the reduction method (3.5) is recommended up to about $n = 40$, and the algorithm (3.4) for $n > 40$.

IV. VECTOR TOEPLITZ MATRICES

We define *vector Toeplitz matrices* as rectangular matrices whose elements are vectors and whose diagonals consist of like elements, except for the vector elements in the last row, which may be obtained by omitting the last components of corresponding full vector elements. Thus a vector Toeplitz matrix has the form

$$A(v) = \begin{bmatrix} v_0 & v_1 & v_2 & \cdots & v_r \\ v_{-1} & v_0 & v_1 & \cdots & v_{r-1} \\ v_{-2} & v_{-1} & v_0 & \cdots & v_{r-2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ v_{1-\ell} & v_{2-\ell} & v_{3-\ell} & \cdots & v_{r+1-\ell} \\ w_{-\ell} & w_{1-\ell} & w_{2-\ell} & \cdots & w_{r-\ell} \end{bmatrix}. \tag{4.1}$$

If we define the elements in (4.1) by

$$v_k = \begin{bmatrix} a_{kp} \\ a_{kp-1} \\ a_{kp-2} \\ \cdots \\ a_{kp-p+1} \end{bmatrix}; \quad w_k = \begin{bmatrix} a_{kp} \\ a_{kp-1} \\ a_{kp-2} \\ \cdots \\ a_{kp+\ell p-n} \end{bmatrix}, \tag{4.2}$$

where ℓ, p, and n are related by

$$\ell = \left[\frac{n}{p}\right], \tag{4.3}$$

then $A(v)$ is a block matrix representation of A, where

$$A = \begin{bmatrix} a_0 & a_p & a_{2p} & \cdots & a_{rp} \\ a_{-1} & a_{p-1} & a_{2p-1} & \cdots & a_{rp-1} \\ a_{-2} & a_{p-2} & a_{2p-2} & \cdots & a_{rp-2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{-p} & a_0 & a_p & \cdots & a_{(r-1)p} \\ a_{-p-1} & a_{-1} & a_{p-1} & \cdots & a_{(r-1)p-1} \\ a_{-p-2} & a_{-2} & a_{p-2} & \cdots & a_{(r-1)p-2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{-2p} & a_{-p} & a_0 & \cdots & a_{(r-2)p} \\ a_{-2p-1} & a_{-p-1} & a_{-1} & \cdots & a_{(r-2)p-1} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{-\ell p} & a_{-(\ell-1)p} & a_{-(\ell-2)p} & \cdots & a_{(r-\ell)p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{-n} & a_{p-n} & a_{2p-n} & \cdots & a_{rp-n} \end{bmatrix}. \tag{4.4}$$

In order that the inversion problem be meaningful, we assume that $r \geq n$. We shall present an algorithm that transforms the submatrix

$$\begin{bmatrix} a_0 & a_p & a_{2p} & \cdots & a_{np} \\ a_{-1} & a_{p-1} & a_{2p-1} & \cdots & a_{np-1} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{-n} & a_{p-n} & a_{2p-n} & \cdots & a_{n(p-1)} \end{bmatrix} \tag{4.5}$$

of A into an upper triangular form. Instead of using "minus" and "plus" interations as in Section II, we introduce an auxiliary matrix

$$B = \begin{bmatrix} b_0 & b_1 & b_2 & \cdots & b_r \\ b_{-1} & b_0 & b_1 & \cdots & b_{r-1} \\ b_{-2} & b_{-1} & b_0 & \cdots & b_{r-2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ b_{1-\ell} & b_{2-\ell} & b_{3-\ell} & \cdots & b_{r+1-\ell} \end{bmatrix}. \tag{4.6}$$

The triangularization of (4.5) is achieved in n recursive steps, transforming $A = A^{(0)}$, $B = B^{(0)}$ successively into $A^{(1)}$, $B^{(1)}$, $A^{(2)}$, $B^{(2)}$, ..., $A^{(n-p)}$, $B^{(n-p)}$; $A^{(n-p+1)}$, $A^{(n-p+2)}$, ..., $A^{(n)}$. In each transformation $A^{(i)} \Rightarrow A^{(i+1)}$, the rows 0, ..., i remain unchanged, and the elements to become new zeros are

$$a^{(i+1)}_{(p-1)i-j} = 0 \qquad (j = 1, 2, \ldots, \min(p-1, n-i)),\ a^{(i+1)}_{-(p+i)} = 0. \tag{4.7a}$$

Thus the matrix $A^{(i)}$ contains i zeros in each row $j \geqq i$ ($j = 0, 1, \ldots, n$). The matrix $B^{(i)}$ contains the element a_0 on the diagonal, and $b^{(i)}_1 = \ldots = b^{(i)}_i = 0$ ($i \geq 1$) are zero elements. The algorithm for the not identically zero elements is as follows. Define $k^{(\tau)}$, $a^{(0)}_s$, and $b^{(0)}_k$ by

$$n - p < pk^{(\tau)} + \tau \leq n \qquad (k^{(\tau)},\ \tau \text{ positive integers});$$

$$a^{(0)}_s = a_s \qquad (-n \leq s \leq rp);$$

$$b^{(0)}_k = a_{kp} \qquad (-\ell < k \leq r).$$

Then

$$a^{(i+1)}_{kp-i-j} = a^{(i)}_{kp-i-j} - \frac{a^{(i)}_{(p-1)i-j}}{a^{(i)}_{(p-1)i}} a^{(i)}_{kp-i} \tag{4.7b}$$

$$(j = 1, 2, \ldots, \min(p-1; n-i);\ k = -k^{(i+j)}, \ldots, -2, -1;\ i+1, \ldots, r);$$

$$a^{(i+1)}_{(k-1)p-i} = a^{(i)}_{(k-1)p-i} - \frac{a^{(i)}_{-p-i}}{a_0} b^{(i)}_k \tag{4.7c}$$

$$(k = -k^{(p+i)}, \ldots, -2, -1;\ i+1, i+2, \ldots, r);$$

$$b^{(i+1)}_k = b^{(i)}_k - \frac{b^{(i)}_{i+1}}{a^{(i+1)}_{(p-1)(i+1)}} a^{(i+1)}_{kp-(i+1)},\ b^{(i+1)}_0 = a_0 \tag{4.7d}$$

$$\left(k = -\left[\frac{n-p-i-1}{p}\right], \ldots, -2, -1;\ i+2, i+3, \ldots, r;\ i \leq n-p\right).$$

These recurrence relations are used for $i = 0, 1, 2, \ldots, n-1$; however, (4.7d) can be terminated at $i = n - p - 1$.

The final matrix $A^{(n)}$ is of upper triangular form

$$A^{(n)} = \begin{bmatrix} a_0^{(0)} & a_p^{(0)} & a_{2p}^{(0)} & \cdots & a_{np}^{(0)} & \cdots & a_{rp}^{(0)} \\ 0 & a_{p-1}^{(1)} & a_{2p-1}^{(1)} & \cdots & a_{np-1}^{(1)} & \cdots & a_{rp-1}^{(1)} \\ 0 & 0 & a_{2p-2}^{(2)} & \cdots & a_{np-2}^{(2)} & \cdots & a_{rp-2}^{(2)} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & a_{np-n}^{(n)} & \cdots & a_{rp-n}^{(n)} \end{bmatrix}. \tag{4.8}$$

To compare the efficiency of (4.7) with the ordinary Gaussian elimination process, we determine the number of multiplications necessary to triangularize (4.5). From (4.8) there follows for $r = n$ that the transformation $A^{(i-1)} \Rightarrow A^{(i)}$ involves only the $(p+1)n - i + 1$ elements $a_{-n}^{(i)}, a_{-n+1}^{(i)}, \ldots, a_{np-i}^{(i)}$. Of these, by (4.7b, c), ip elements are zero and need not be calculated. Therefore, each such transformation requires $(p+1)n - (p+1)i + 1$ multiplications if $i = 1, 2, \ldots, n-p+1$. In addition, the transformations $B^{(i-1)}$ to $B^{(i)}$, by (4.7d), involve the $\left[\frac{n-p-i+1}{p}\right] + n + 1 = \left[\frac{n-i+1}{p}\right] + n$ elements $b_k^{(i)}$, $-\left[\frac{n-i+1}{p}\right] < k \le n$, if $r = n$. Of these elements, $i + 1$ are known, namely, $b_0^{(i)} = a_0$, $b_1^{(i)} = \ldots = b_i^{(i)} = 0$, and need not be computed. Therefore each transformation (4.7d) requires $\left[\frac{n-i+1}{p}\right] + n - i - 1$ multiplications if $i = 1, 2, \ldots, n-p+1$. The total number of multiplications to obtain $A^{(n-p+1)}$ and $B^{(n-p+1)}$ is no greater than*

$$\sum_{i=1}^{(n-p+1)} \left[(p+1)n - (p+1)i + 1 + \frac{n-i+1}{p} + n - i - 1\right] = \\ (n-p+1)\frac{(p+1)^2(n+p-2)+2}{2p}. \tag{4.9}$$

The matrix $A^{(n-p+1)}$ for $r = n$ has the form

$$\begin{bmatrix} a_0^{(0)} & a_p^{(0)} & \cdots & \cdots & a_{np}^{(0)} \\ 0 & a_{p-1}^{(1)} & \cdots & \cdots & a_{np-1}^{(1)} \\ 0 \;\cdots\; 0 & & a_{(n-p+1)(p-1)}^{(n-p+1)} & \cdots & a_{np-n+p-1}^{(n-p+1)} \\ 0 \;\cdots\; 0 & & a_{(n-p+1)p-n}^{(n-p+1)} & \cdots & a_{np-n}^{(n-p+1)} \end{bmatrix}. \tag{4.10}$$

---

*$\left[\frac{n-i+1}{p}\right] \le \frac{n-i+1}{p}$.

Therefore, to bring (4.10) into the triangular form $A^{(n)}$ requires the same number of multiplications as a square matrix of order p, i.e.,

$$\frac{p(p-1)(2p-1)}{6} \tag{4.11}$$

multiplications. Addition of (4.11) to (4.9) yields the upper limit for the total number of multiplications required to transform A into $A^{(n)}$ $(r = n)$, namely,

$$(n-p+1)\frac{(p+1)^2(n+p-2)+2}{2p} + \frac{p(p-1)(2p-1)}{6} \tag{4.12a}$$

$$\approx \frac{p}{2}(n^2-p^2) + \frac{p^3}{3} = \frac{n^2 p}{2} - \frac{p^3}{6}. \tag{4.12b}$$

The same transformation using the Gaussian algorithm requires

$$\frac{n}{6}(n+1)(2n+1) \approx \frac{n^3}{3} \tag{4.13}$$

multiplications. The savings factor, the ratio (4.13) to (4.12), is asymptotically $\frac{2}{3}\frac{n}{p}$. If $p \geq n+1$, (4.7) reduces the Gaussian elimination method. If $p = 1$ and $r = n$, (4.7) is just another representation for the algorithm (2.6b).

To solve

$$Ax = 0 \quad (r = n+1) \tag{4.14}$$

for x we have to increase the number given by (4.12a) by the following numbers of multiplications:

n to calculate $a^{(i)}_{n+1-i}$ $(i = 1, \ldots, n)$;

n - p to calculate $b^{(i)}_{n+1}$ $(i = 1, \ldots, n-p)$;

$\frac{n(n+1)}{2}$ to perform the back substitution in $A^{(n)}$.

These numbers combine to a total of

$$\frac{n}{2}(n+5) - p \tag{4.15}$$

additional multiplications.

We realize that we have not included an error and stability analysis for the algorithms. No simple operator representation in matrix form has been found for the description of the algorithms; it would facilitate these analyses. We hope, however, the answers to these problems will be solved and the new algorithms will prove themselves useful in practical applications.

ACKNOWLEDGMENTS

David L. Phillips and Ibrahim K. Abu-Shumays checked the formulas in this paper. Dave Phillips also performed numerical calculations.

REFERENCES

1. Norbert Wiener, Extrapolation, Interpolation, and Smoothing of Stationary Time Series, Appendix B by Norman Levinson, John Wiley & Sons, Inc., New York (1949), pp. 129-139.

2. William F. Trench, An Algorithm for the Inversion of Finite Toeplitz Matrices, J. Soc. Indust. Appl. Math., 12, 515-522 (Sept 1964).

3. William F. Trench, An Algorithm for the Inversion of Finite Hankel Matrices, J. Soc. Indust. Appl. Math., 13, 1102-1107 (Dec 1965).

4. William F. Trench, Weighting Coefficients for the Prediction of Stationary Time Series from the Finite Past, SIAM J. Appl. Math., 15, 1502-1510 (Nov 1967).